OpenCUDA+MPI

A Framework for Heterogeneous GP-GPU Distributed Computing

Kenny Ballou

March 15, 2013

Overview

Implementation Details

- Arch Linux
- Python
- Salt
- CUDA
- (Open)MPI

Arch Linux

- Core Tenet: Minimalism
- Small
- Lightweight
- Familiarity

Python

- Development Speed: Expressive and Readable
- Fast Enough
  - Written in `C/C++`
- Where slow, allows use of `C/C++` code
- `pyCUDA`: Deferred CUDA kernel compilation

Salt

More than just NaCl

Provisioning tool for managing infrastructure

- Allows for "Push" based state changes
- Remote Execution
- Simplicity
- Fast

Tasks Completed

- Learn and Write "Hello World" Programs with `MPI`, `mpi4py`, and `pyCUDA`
- Write/ Modify Node Build Script
  - Add Configuration
- Start adding 'Salt'

Moving Forward

- Finish Creating Salt States and Configuration
- Stabilize Environment
- Continue Learning `MPI` and `pyCUDA`
- Develop Framework

Problems Encountered

- Power Requirements
- Node Stability/ Package (Mis-)Management
- NFS Share `/home` performance
- Time

Foreseeable Problems and Challenges

- Data Distribution
  - NFS Share will not suffice
- Fault Tolerance
- Optimizing Resources and Utilization

Later Tasks and Potential Solutions

- Request(ing) more suitable and stable power
- Hadoop Distributed Filesystem

OpenCUDA+MPI

A Framework for Heterogeneous GP-GPU Distributed Computing

Kenny Ballou

March 15, 2013